CS 520

Theory and Practice of Software Engineering
Fall 2018

Testing Software for Fairness

October 2, 2018

Coming up

- Guest lecture tomorrow (Wednesday), October 3 4-5PM in CS 151
  - Please attend.
    (if you have a time conflict – we'll post a video)
- No class on Thursday, October 4
- Extra credit assignment posted.

CS 520
Extra Credit
Debugging Study

This semester, we will offer an (optional) extra credit assignment involving a study on debugging Java programs.

The extra-credit assignment will be done in-person, during a two-hour one-on-one session. You will be presented with Java code written by someone else and asked to perform coding tasks using modern development tools.

How to sign up:

If you wish to participate in this extra credit assignment, you have to email bjohnson@cs.umass.edu with the subject:
`CS 520 Extra Credit scheduling request`
with the body of the email containing your name and a short message saying you would like to participate in the extra credit assignment. You will receive an email back with instructions on how to schedule your two-hour session.

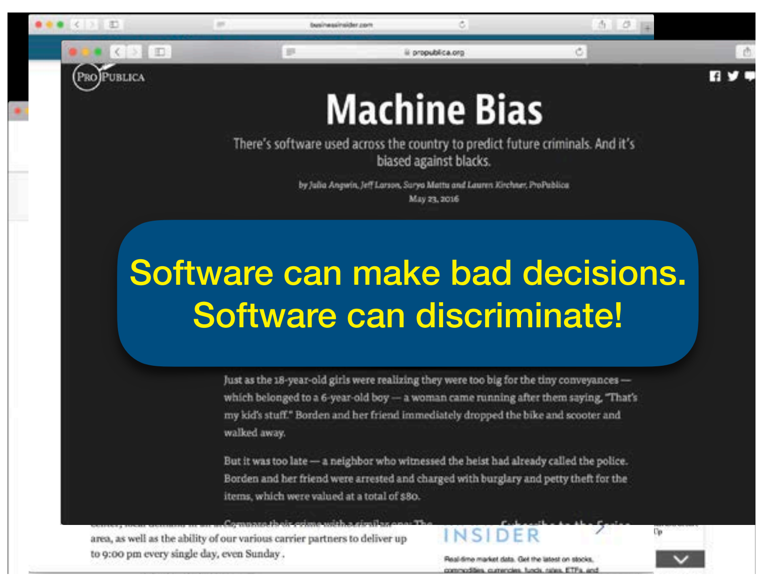
This extra credit assignment will be done on a first-come-first-served basis. While we anticipate that we will be able to accomodate everyone who wants to participate, if too many people wish to participate, we may stop administering the assignment after a certain point. Once you have scheduled a session, you are guaranteed to get a chance to participate.

Point value:

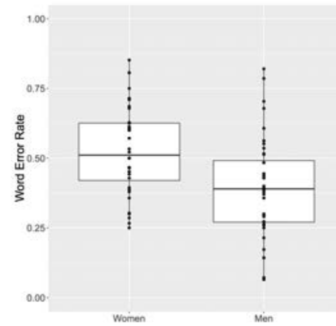
This assignment will be worth up to 2 points on your final grade. For reference, each of the in-class exercises is worth 7.5 points, so completing this extra credit is like a 26.7% boost to one in-class exercise's grade.

Assignments

- Homework 1 (due Oct 16):
  https://people.cs.umass.edu/~brun/class/2018Fall/CS520/hw1.pdf
- Final project assignment:
  https://people.cs.umass.edu/~brun/class/2018Fall/CS520/finalProject.pdf
- End of class today, time to discuss groups



YouTube Automatic captions



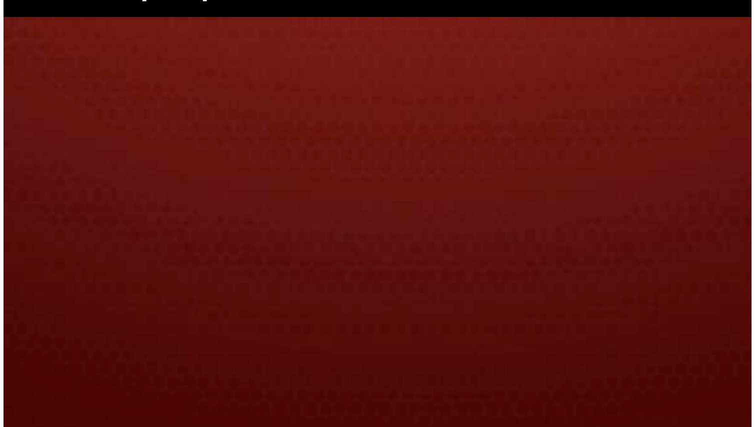
Rachael Tatman, "Gender and Dialect Bias in YouTube's Automatic Captions" in 2017 Workshop on Ethics in Natural Language Processing

YouTube Automatic captions

Rachael Tatman, "Gender and Dialect Bias in YouTube's Automatic Captions" in 2017 Workshop on Ethics in Natural Language Processing

Joy Buolamwini
https://www.ted.com/talks/joy_buolamwini_how_i_m_fighting_bias_in_algorithms

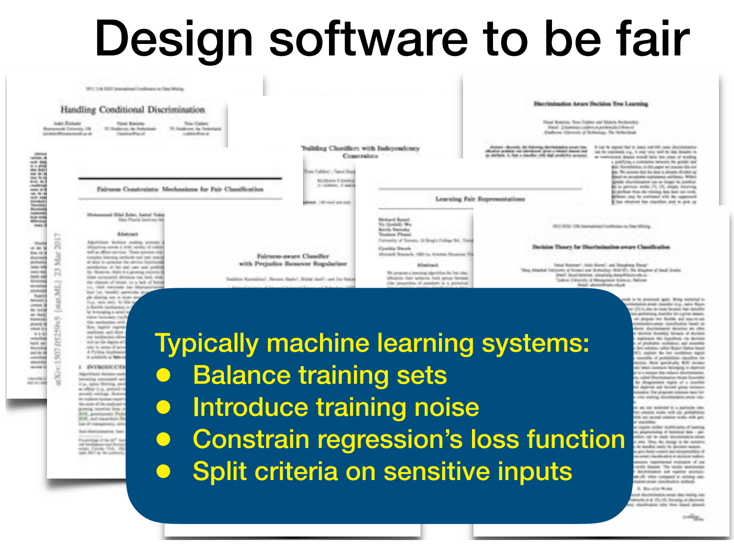
how people want to use vision software

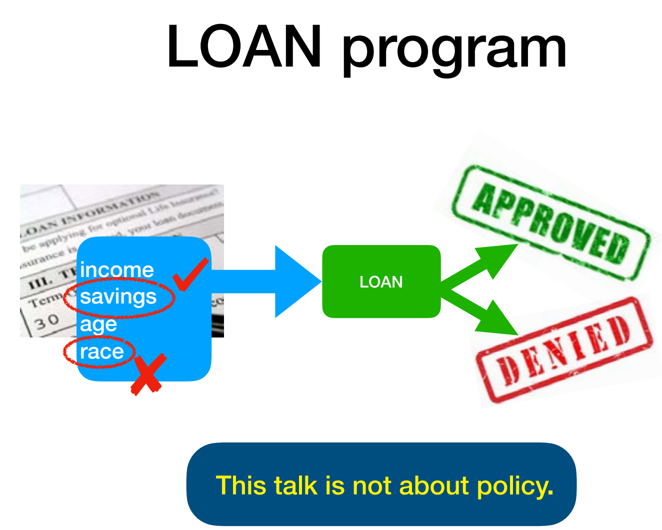


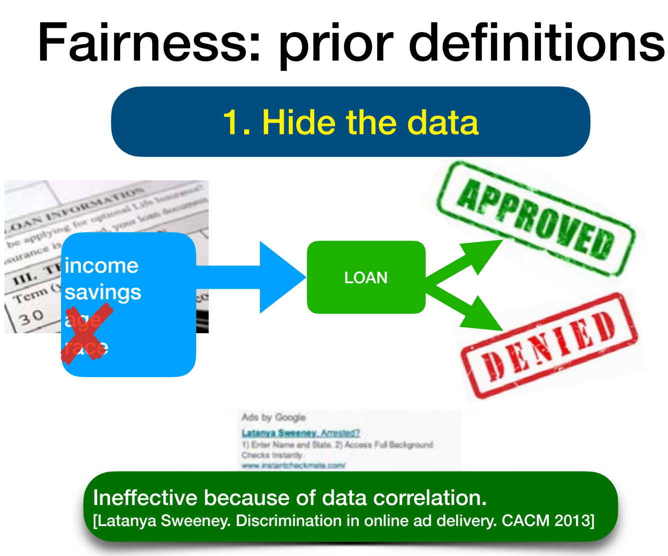
today's goals

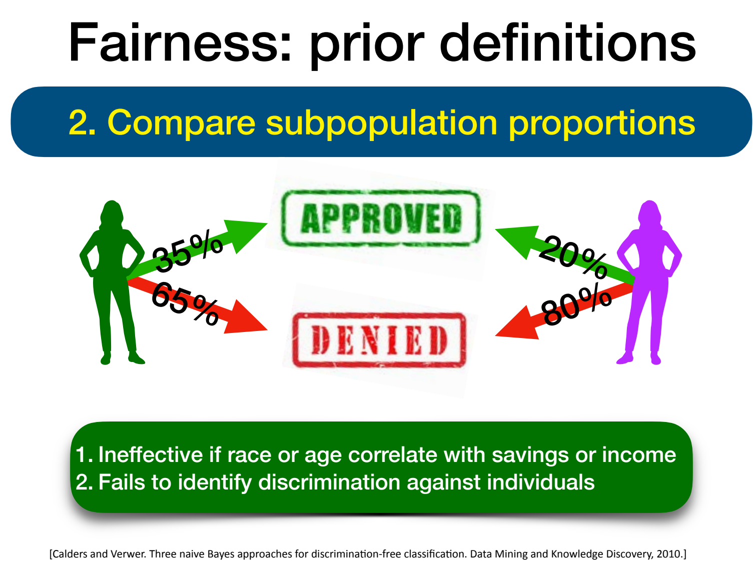
Define software discrimination.

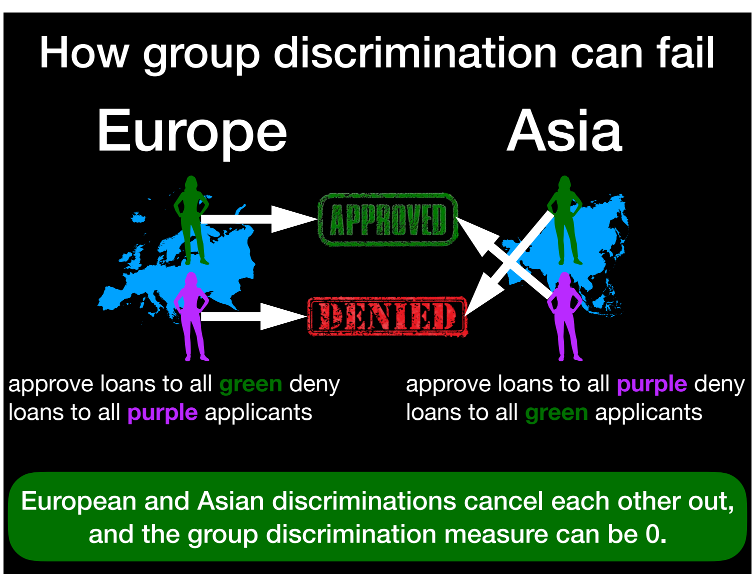
Operationalize measuring discrimination through causal software testing.

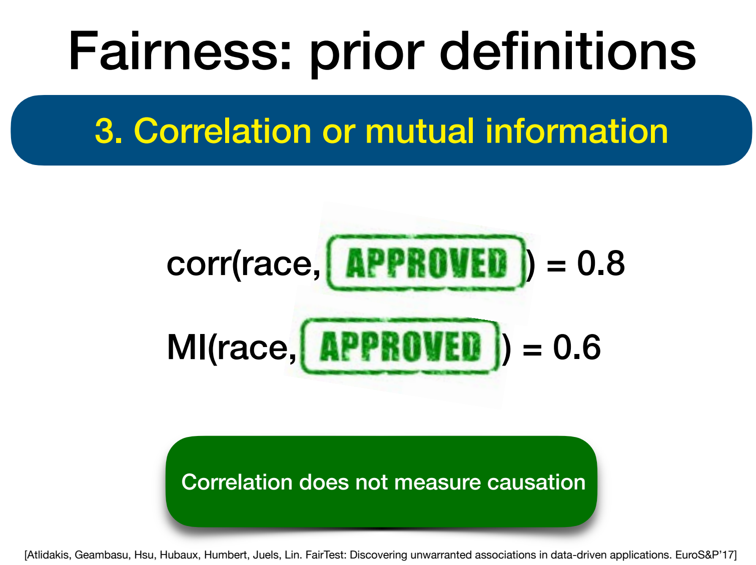

Design software to be fair
Typically machine learning systems:
• Balance training sets
• Introduce training noise
• Constrain regression's loss function
• Split criteria on sensitive inputs


LOAN program
income
savings
age
race
LOAN
APPROVED
DENIED
This talk is not about policy.


Fairness: prior definitions
1. Hide the data
income
savings
LOAN
APPROVED
DENIED
Ads by Google
Latanya Sweeney, Arrested?
Ineffective because of data correlation.
[Latanya Sweeney. Discrimination in online ad delivery. CACM 2013]


Fairness: prior definitions
2. Compare subpopulation proportions
35%
65%
APPROVED
DENIED
20%
80%
1. Ineffective if race or age correlate with savings or income
2. Fails to identify discrimination against individuals
[Calders and Verwer. Three naive Bayes approaches for discrimination-free classification. Data Mining and Knowledge Discovery, 2010.]


How group discrimination can fail
Europe
Asia
APPROVED
DENIED
approve loans to all green deny loans to all purple applicants
approve loans to all purple deny loans to all green applicants
European and Asian discriminations cancel each other out, and the group discrimination measure can be 0.


Fairness: prior definitions
3. Correlation or mutual information
corr(race, APPROVED) = 0.8
MI(race, APPROVED) = 0.6
Correlation does not measure causation
[Atlidakis, Geambasu, Hsu, Hubaux, Humbert, Juels, Lin. FairTest: Discovering unwarranted associations in data-driven applications. EuroS&P'17]

What is fairness?

Sensitive inputs should not affect software behavior.

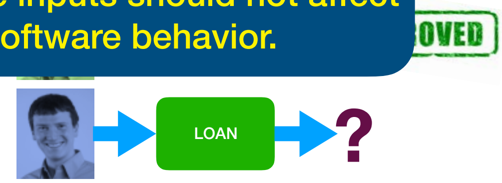
We want to measure causality!

[Judea Pearl. Causal inference in statistics: An overview. Statistics Surveys 2009]

causal testing

Sensitive inputs should not affect software behavior.

hypo… testing:



causal testing

causal testing

Themis

automated test-suite generator

How much does my software discriminate with respect to …?

Does my software discriminate more than 10% of the time, and against what?

Themis generates a test suite or can use a manually written one

http://fairness.cs.umass.edu

discrimination measures

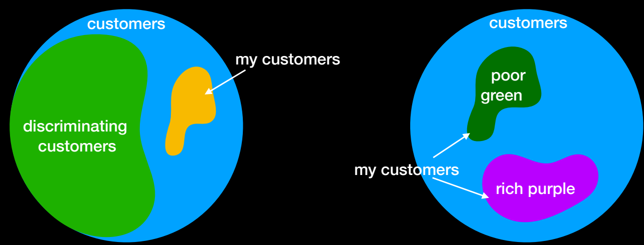
causal discrimination

LOAN() $\stackrel{?}{=}$ LOAN ()

group discrimination

apparent discrimination (causal or group)

apparent discrimination

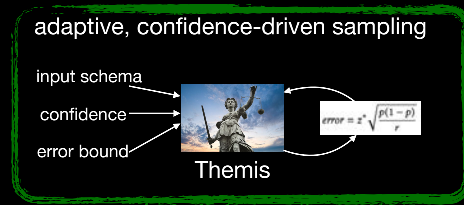


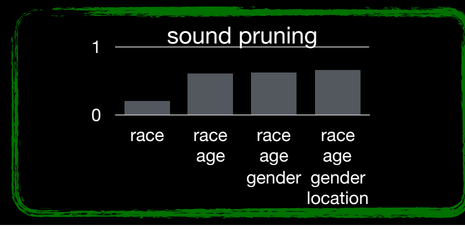
Software may discriminate, but not for a given set of customers

Fair software may appear to discriminate
(e.g., Amazon same-day delivery)

★ Apparent discrimination can be group or causal, measured on a given test suite or operational profile.

How does Themis work?





Evaluation

Eight open-source decision systems trained on two public data sets

discrimination-aware logistic regression	[88]
discrimination-aware decision tree	[40]
discrimination-aware naive Bayes	[18]
discrimination-aware decision tree	[91]
naive Bayes	scikit-learn
decision tree	scikit-learn
logistic regression	scikit-learn
SVM	scikit-learn

- Census income dataset:
  financial data
  45K people
  income > $50K?
- Statlog German credit dataset:
  credit data
  1K people
  “good” or “bad” credit?

findings

Group discrimination is not enough.

More than 11% of the individuals had the output flipped just by altering the individual’s gender.

Decision tree trained not to group discriminate against gender causal discriminated against gender: 0.11.

findings

Trying to avoid group discrimination may introduce other discrimination.

Training a decision tree not to discriminate against gender made it discriminate against race 38.4% of the time.

findings

Pruning is highly effective.

- The more a system discriminates, the more efficient Themis is.
- On average, pruning reduced test suites by 148× for causal and 2,849× for group discrimination. Best improvement was 13,000×.

related work

Ways of measuring discrimination

- CV score [19]
- correlation, mutual information [79]
- Output probability distributions [51]

Discrimination-aware algorithms [18, 40, 88, 91]

Measuring discrimination with manually-written tests [79]

Causal model inference [Maier et al., UAI'13]

Fairness verification [Albarghouthi et al., OOPSLA'17]

what's next?

- Software with complex inputs, such as natural language or photographs and videos.
- What definition is right for what software requirements context?
- Efficiency in testing.

Contributions

http://fairness.cs.umass.edu

- Causality-based definition and method for measuring software fairness
- Themis, an automated test-suite generator for fairness testing
- Provably-sound pruning test-suite reductions
- Evaluation on real-world software, demonstrating Themis' effectiveness

CS 520
Final project description

Final projects will be completed in teams of 4 or 5 students. Each team is responsible for a single project.
You should select a team and a project by **Tuesday, October 9, 2018, 9:00AM EDT**.
Your mid-point check-in will be on **Tuesday, November 13, 2018, 9:00AM EST**.
The final project will be due **Tuesday, December 11, 2018, 11:55 PM EST**.
There are five options for a final project (each team will do one):

1. MSR 2019 Mining Challenge
2. Replication study
3. Model Inference for Inferring Processes
4. EleNa: Elevation-based Navigation
5. Self-defined software engineering research project

MSR 2019 Mining Challenge

The Mining Software Repositories conference runs an annual challenge in which they provide a dataset and ask you to answer research questions about the dataset. Read the description of this year's dataset, research questions, and challenge here:
https://2019.msrconf.org/track/msr-2019-Mining-Challenge#Call-for-Mining-Challenge-Papers

Replication study

A replication study takes an existing research paper, replicates its experiments on the same data, and then extends the experiments to expanding that data set on which the experiments are run. For this project, we highly recommend selecting a paper with publicly available dataset and code to execute the experiments. The project involves a write up describing the process of replicating the experiments, deviations in the achieved results from the original ones reported in the paper, and lessons learned from applying the experiments to new data.

Here is a list of several papers that are good candidates for replication:

1. Automatic generation of oracles for exceptional behaviors from Javadoc comments.

   Paper: https://dl.acm.org/citation.cfm?id=2931061

   Source code: https://github.com/albertogoffi/toradocu

2. SimFix: Automated program repair

   Paper: http://sei.pku.edu.cn/~xiongyf04/papers/ISSTA18a.pdf

   Source code: https://github.com/xgdsmileboy/SimFix

   Dataset: https://github.com/rjust/defects4j